Parameterizing Deformable Surfaces for Monocular 3–D Tracking

Mathieu Salzmann, Slobodan Ilic, Pascal Fua
Computer Vision Laboratory
Ecole Polytechnique Federale de Lausanne (EPFL)
CH-1015 Lausanne, Switzerland
Mathieu.Salzmann,Slobodan.Ilic,Pascal.Fua@epfl.ch
http://cvlab.epfl.ch
Technical Report ID: IC/2005/020

Abstract

We propose a deformable surface parameterization that is generic and lets us automatically build registered shape databases. This allows us to directly derive low-dimensional shape models using a simple dimensionality reduction technique. This addresses one of the biggest difficulties in example-based shape modeling: Building the required database, which is often a difficult and painstaking process.

We incorporate the resulting models into a monocular tracking system that we use to capture the complex deformations of objects such as sheets of papers or expanding balloons from single video sequences.

1 Introduction

While reconstruction and tracking of rigid and articulated objects from video have been widely studied, modeling 3–D deformable surfaces such as those of Fig. 1 remains a challenging problem, especially with a single camera. The problem would be totally under-constrained without an appropriate deformation model, that is one flexible enough to account for all possible configurations of the deformable objects while being controlled by sufficiently few parameters for effective optimization.

Many physically-based models have been proposed, but they seldom incorporate all the required physical knowledge. Even a simple sheet of paper is a complex dynamic system and the models that are popular [9, 10, 11, 12, 5, 7] tend to be oversimplified. They often rely on linear approximations that are poor in the presence of large displacements and deformations such as those the objects shown in Fig. 1 undergo. More realistic non-linear models have been investigated [16, 13]. But, to the best of our knowledge, they are complex enough never to have been tried for monocular shape recovery.

An alternative to physically-based modeling is to create a database of valid shapes as was done for faces [4, 3] or human motion [14] and use a dimensionality reduction technique to derive models with relatively few degrees of freedom. This approach has proved very effective *if* the database can be obtained, which is not a given. For example, in the case of faces [4, 3], its construction was a painstaking process that required precise registration of individual vertices of a generic face model to laser scans.

To alleviate the database generation problem in the case of generic deformable surfaces, we propose a parameterization that lets us automatically generate a wide range of physically valid configurations. We represent surfaces as triangulated meshes whose shape is controlled by a small subset of the angles between the facets. Varying these angles results in the database we need, which contains surfaces of different shapes but identical topologies. We then perform a Principal Component Analysis and approximate surfaces as linear combinations of a small number of principal components. The resulting deformation model has relatively few parameters and, yet, is accurate enough to effectively track deformations.

The contribution of this paper is therefore a surface parameterization that lets us sample the space of valid shapes to build a representative database, which can then be used to derive low-dimensional shape models using a simple dimensionality reduction technique. We will show that these models are effective to capture the deformations of objects such as sheets of papers or expanding balloons from monocular video sequences.

2 Related Work

Capturing surface deformations from a single video stream is acknowledged to be a massively under-constrained problem if one does not limit the range of possible configurations. Existing approaches can be classified into two broad categories: Physically-based methods seek to parameterize the surfaces in terms of the variables of a dynamic system that approximates the real physics, while example-based techniques rely on creating a database of possible shapes from which a low-dimensional model can be learned. We briefly review these two classes of approaches below.

The original snake paper [9] probably is the one that contributed most to popularize physically-based models in the Computer Vision community. The approach was initially strictly 2–D, but was soon extended to 3–D surface modeling, by using deformable superquadrics [15, 6], triangulated meshes [1], or thin-plate splines [10]. Unfortunately, these modeling techniques tend to produce models with too many degrees of freedom for reliable fitting to monocular sequences. An approach to reducing the number of degrees of freedom is to perform modal analysis [12, 5, 7]. The object's behavior is then described by superposing its natural strain and vibration modes. However, this implies linear assumptions that do not hold when the deformations become large. The use of non-linear finite elements has been investigated in the medical imaging and animation communities for volumetric reconstruction and simulation [8, 13]. However, for 3–D surface fitting purposes, such methods have only been demonstrated for recovering relatively simple deformations from range data [16] and require precise knowledge of the object's material properties, which may be hard to obtain.

Because accurately modeling the physics of deformable surfaces is difficult, example-based methods are an attractive alternative. They involve creating a database of representative shapes and using them in conjunction with a statistical dimension reduction technique to learn a model with comparatively few degrees of freedom. For example, the work of Blanz and Vetter on facial shape recovery [4] and animation [3] relies on a deformable face model built in this way. The database is made of 3–D meshes that were fitted to laser scans and aligned so that specific vertices always correspond to the same facial features. The shape model is learned by performing Principal Component Analysis on the vectors

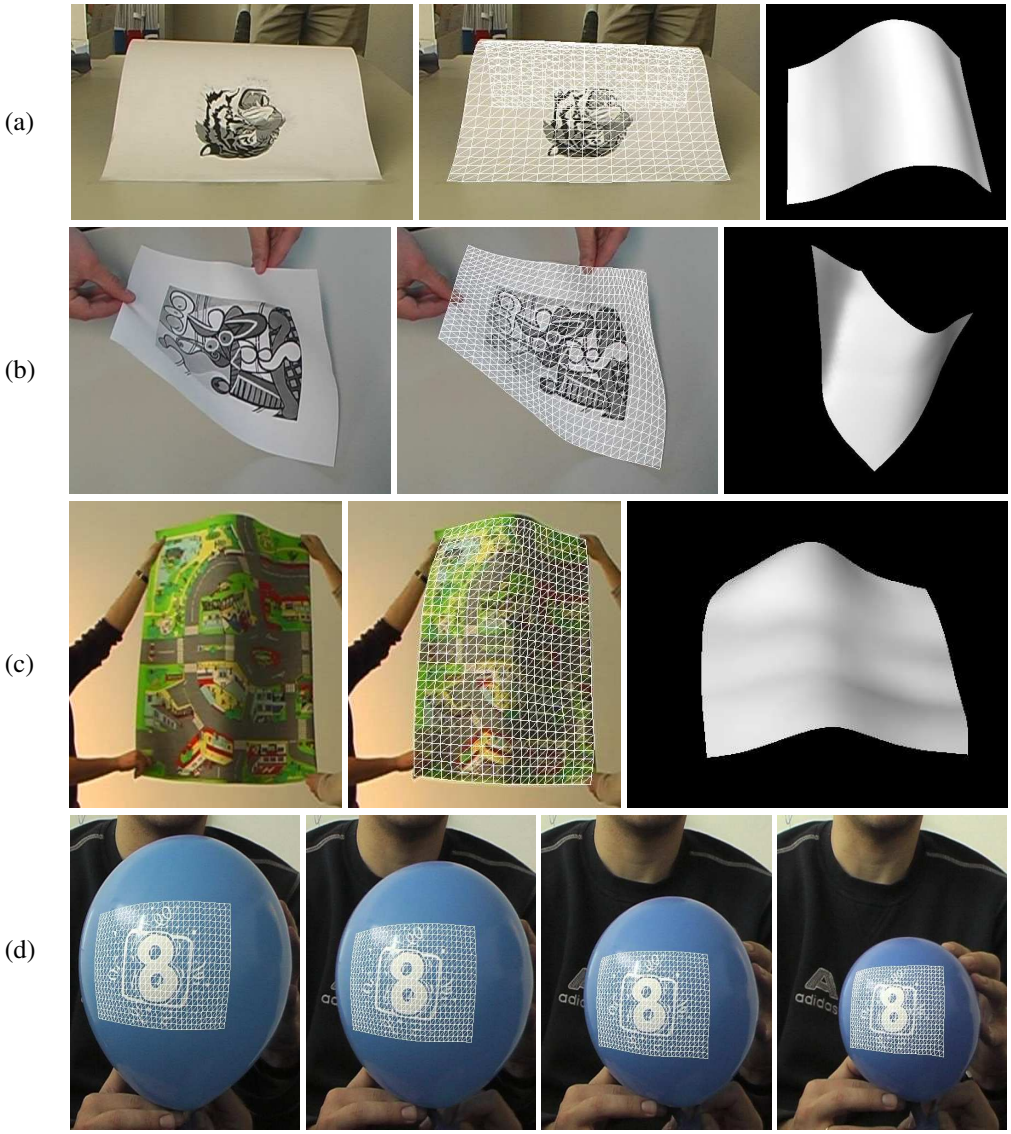


Figure 1: Tracking deforming surfaces in monocular videos. (a, b, c) On each row, we show one image of the original sequence, the deformed mesh projected on the sequence, and a shaded side view. (d) The shape of a deflating balloon is tracked throughout a sequence. Note that the mesh shrinks along with the balloon.

formed by concatenating the 3–D coordinates of the mesh vertices and only retaining the most significant components. Similar approaches were successfully used to derive articulated motion models [2, 14]. However, gathering and registering enough examples to build a meaningful database again represented a very significant amount of work.

The difficulties involved in creating the databases have limited the spread of these

example-based approaches to other applications. This is the issue we address here in the case of generic deformable surfaces.

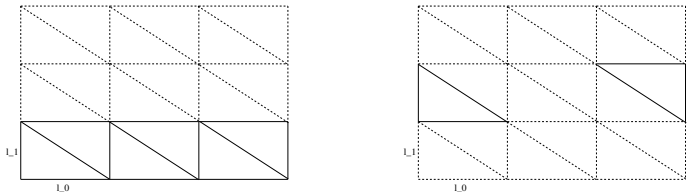
3 Creating the Deformation Model

Our approach is built on one key insight: A surface that deforms and stretches can always be resampled so that it can be represented by a triangulated mesh whose vertices are equidistant. In other words, with no loss of generality, we can learn all the possible shapes from a set of *inextensible* meshes. These are meshes whose edges are constrained to retain their length and whose only degrees of freedom are the angles between the facets.

In this section, we show that only a small subset of these angles need be specified to fully constrain the shape. As a result, we can represent the whole set of valid shapes using these parameters and therefore produce a representative set of shapes by randomly sampling them. We exploit this to create a database of meshes that are naturally registered to one another and are therefore directly amenable to dimensionality reduction using a simple technique such as PCA.

3.1 Parameterizing Inextensible Meshes

We represent surfaces as triangulated meshes whose vertices lay on an $M \times N$ rectangular grid, such as the one depicted by Fig. 2. If we constrain the lengths l_0 and l_1 to be the same for all horizontal and vertical edges, such a mesh has far fewer degrees of freedom than the $3 \times M \times N$ ones required to individually specify the vertex coordinates.



(a) (b)

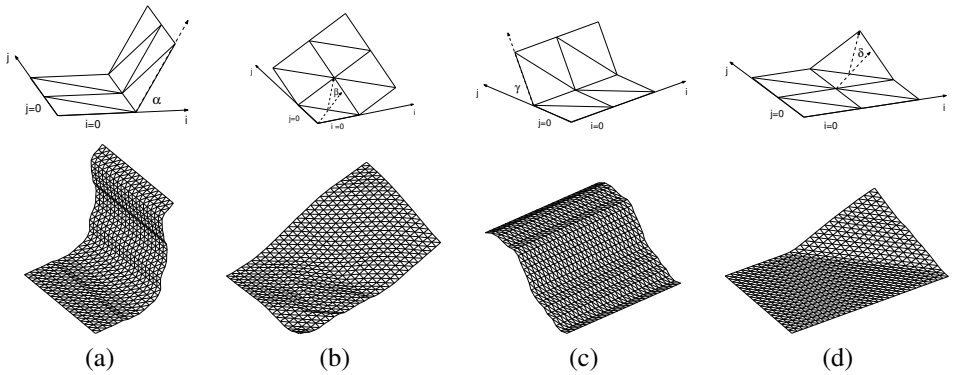
Figure 2: Building a model. (a) The bottom row of the mesh is first built from left to right by setting the angle between a facet and its neighbor. (b) For each consecutive row, only two angles need be set, one for the first facet and one for the last one.

More specifically, as shown in Fig. 2 (a), we can specify their 3–D shape starting from the bottom row. The 3–D orientation of the bottom left facet is given by two rotations around the x and y axes. The position of each successive facet is then recursively defined by the rotation angle around the edge it shares with the previous one. Once the first row has been specified in this manner, it can be shown that there are only two degrees of freedom left for each successive row. We therefore proceed row by row and fix those degrees of freedom by specifying the rotation angles of the two facets drawn in bold in Fig. 2 (b), around the edge it shares with the facet below for the lower-left one and around its neighbor to the left for the upper-right one. The 3–D coordinates of all vertices in the row can then be recursively computed as the intersection of three spheres of radii l_0, l_1, and $\sqrt{l_0^2 + l_1^2}$ centered at vertices whose coordinates have already been computed.

In short, given the horizontal and vertical lengths l_0 and l_1, an inextensible surface can be parameterized in terms of four sets of angles:

- α_i , $0 \leq i < M-1$: Orientation of the left triangle of column i in the first row.
- β_i , $0 \leq i < M-1$: Orientation of the right triangle of column i in the first row.
- γ_j , $1 \leq j < N-1$: Orientation of the left triangle of row j in the first column.
- δ_j , $1 \leq j < N-1$: Orientation of the right triangle of row j in the last column.

The tow row of Fig. 3 illustrates the effect of varying these angles individually.



(a) (b) (c) (d)

Figure 3: Top row: Setting a single angle to a non zero value for one of the α_i, β_i, γ_j, δ_j. Bottom row: Setting all the angles to non zero values independantly for the α_i, β_i, γ_j, δ_j.

3.2 Building the Shape Database

We create a database of randomly deformed meshes by letting the angles discussed above vary randomly between two fixed bounds. To cover the whole range of shapes, we could sample the space spanned by the $\{\alpha_0,\ldots,\alpha_{M-2},\beta_0,\ldots,\beta_{M-2},\gamma_1,\ldots,\gamma_{N-2},\ldots,\delta_1,\ldots,\delta_{N-2}\}$. As this would still require a huge number of samples, we chose instead to sample each one of the four sets of angles independently to produce shapes such as those of the bottom row of Fig. 3. As will be shown in Section 3.3, this does not reduce the generality of the approach.

In practice, we use $M = 30$, $N = 20$, and choose angles in the range $[-\pi/6, \pi/6]$. These values yield surfaces with potentially large global curvature but that remain locally smooth, such as the ones of Fig. 5. We generate 50 random meshes for each set of angles. As will be discussed in the next section, these meshes are to be used to perform Principal Component Analysis. To guarantee that the resulting components are as symmetric as possible, we symmetrize our 50 samples as follows. When sampling the $\{\beta_i\}$ and $\{\delta_j\}$ angles, we also include the symmetrical counterparts to our samples with respect to the x and y coordinates, which results in a total of 200 meshes being added to the database for each of these sets of angles. In the case of the $\{\alpha_i\}$ and $\{\gamma_j\}$ angles, a single symmetry suffices, resulting in a total of 100 meshes being added to the database for each of these sets of angles. Finally, we end up with a database containing 600 mesh examples for a total of $2 \times (M-1) + 2 \times (N-2) + 1 = 95$ degrees of freedom.

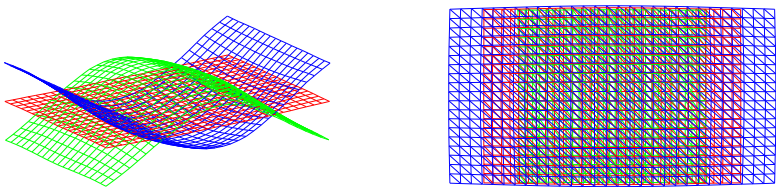
3.3 Principal Component Analysis

To further reduce the number of parameters required to represent our deformable surfaces, we use Principal Component Analysis (PCA). Since all database meshes have the same topology, we form a $3 \times M \times N$ vector for each one by concatenating the coordinates of its vertices. By running PCA on these vectors and retaining only the first $N_c << 3MN$ principal components, we can approximate the vector of coordinates of any mesh as

$$S = \bar{S} + \sum_{k=1}^{N_c} w_k S_k \ , \tag{1}$$

where $\bar{S}$ is the vector corresponding to an undeformed mesh, the S_k are the principal components or modes, and the w_k are weights that specify the surface shape. In other words, the shape of a mesh can now be expressed as a function of the state vector $\Theta = \{w_1, ..., w_{Nc}\}$. In practice, $3MN = 1800$ and we take N_c to be at most 50.

Fig. 4 depicts the influence of two of the most significant components. Changing the weight associated to the first produces bending and, to the second, extension. This is an important feature of our approach: Even though the database contains only inextensible meshes, the resulting components allow the modeling of shrinkage and extension, a fact that we will exploit in Section 5.2.



(a) (b)

Figure 4: Modal behavior. In both figures, $\bar{S}$, the average mesh, is shown in red. The other two are obtained by taking a single w_k to be non zero. A positive value of that w_k yields the green mesh and a negative one the mesh shown in blue. (a) Bending for $k = 9$. (b) Extension for $k = 5$.

Recall from Section 3.2 that we created the database by independently, as opposed to simultaneously, varying the four sets of α, β, γ, and δ angles. Arguably, this could fail to cover all possible deformations and result in principal components unable to describe some configurations. To disprove this, we generated a number of synthetic meshes such as the ones of Fig. 5 by simultaneously randomizing all the angles and verified that we can use our $N_c = 50$ principal components to accurately fit the resulting shapes.

Figure 5: Fitting surfaces created by varying all sets of angles simultaneously. The original shapes are shown as shaded, while the fitted ones are displayed as wireframes.

4 Optimization Framework

As discussed in Section 3.3, the shape of the mesh is controlled by a state vector Θ of weights associated to the principal components. We use the image data to write n_{obs} observation equations of the form

$$Obs^{type_i}(\mathbf{x}_i, \Theta) = \varepsilon_i \ , \ 1 \leq i \leq n_{\text{obs}} \ , \tag{2}$$

where Obs^{type_i} is a differentiable objective function associated to a particular type of image data, $\mathbf{x}_i$ a data point, and ε_i is an error term. Here we consider the functions Obs^{corr} and Obs^{edge} derived from point correspondences and edge information respectively:

- Correspondences. We find interest points in the first of consecutive pairs of images and compute correspondences in the second. Given a couple $\mathbf{u}_i = (p_i^1, p_i^2)$ of corresponding points found in this manner , we define an $Obs^{corr}(\mathbf{u}_i, \Theta)$ as follows: We back-project p_i^1 to the 3–D surface and reproject it to the second image. We then take $Obs^{corr}(\mathbf{u}_i, \Theta)$ to be the Euclidean distance in the image plane between this reprojection and p_i^2.
- Boundary and occluding contours. We project the target object boundaries into the image. We then sample the projections and look for the closest image edge-pixel in the normal direction. We take Obs^{edge} to be the Euclidean distance between the projection and an edge-pixel. We handle occluding contours similarly. We use OpenGL visibility computation and hidden surface removal techniques to find mesh edges that correspond to occluding contours. We then sample these edges and evaluate Obs^{edge} as discussed above.

We will show in the results section that this combination suffices to fully constrain the surface's shape.

As we saw in Section 3.3, a linear combination of principal components can result in a mesh that expands or shrinks. To model surfaces that do not stretch, such as a piece of paper, we force edge lengths to remain constant by introducing a penalty term

$$E_D \quad = \quad \sum_{i=1}^{Nvert} \sum_{v_j \in N(v_i)} (\left\| v_i - v_j \right\| - L_{i,j})^2 \ , \tag{3}$$

where v_i is a vertex of the mesh, $N(v_i)$ represents the set of all its neighbors, and $L_{i,j}$ is the initial edge length. Finally, we take the global objective function E we minimize to be

$$E = \frac{1}{2} \sum_{i=0}^{n_{obs}} w_{type_i} \left\| Obs^{type_i}(\mathbf{x}_i, \Theta) \right\|^2 + w_{ext} E_D \ , \tag{4}$$

where the w_{type_i} are weights associated to particular observation types and designed so that the derivatives of all observations are of commensurate magnitude, and w_{ext} is a user defined weight. A small, or zero, w_{ext} allows the mesh to stretch or shrink.

5 Results

Here we demonstrate the robustness of our approach for tracking objects undergoing large deformations. We provide the corresponding videos as supplementary material.

5.1 Inextensible Surfaces

We first applied our method to tracking deformable but inextensible surfaces in monocular sequences. Not only does the Θ state vector that controls the shape contain relatively few parameters, but we do not need a regularization or smoothing term. Simply keeping the number of principal components we use low is enough to enforce smoothnes. However, we had to fix some coordinates of the meshes to avoid ambiguities due to the chosen viewpoints.

Fig. 6 depicts the tracking of a piece of paper starting from an undeformed position. Even though there is texture at only one place on the paper, the whole model deforms correctly. This includes the back of the sheet that is not actually seen in the video. Another deforming sheet of paper is shown in Fig.7. The chosen viewpoint makes it difficult to clearly see the deformation in the first frames. Our algorithm nevertheless retrieves the precise 3–D shape throughout the whole sequence. In both cases, we used 30 principal components.

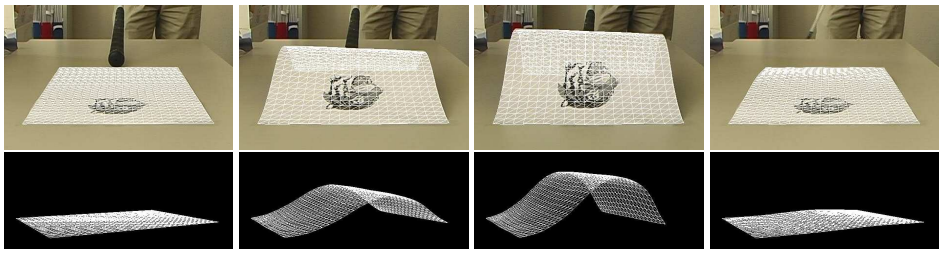


Figure 6: Deforming sheet of paper. Top row: Deformed mesh projected on the original sequence as a wireframe. Bottom row: Deformed mesh shown as a wireframe model seen from a different viewpoint. Note that even the back deforms correctly.

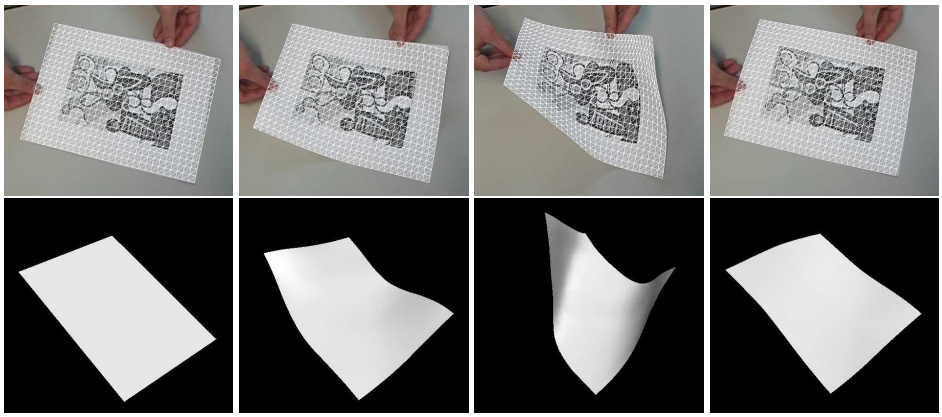


Figure 7: Another deforming sheet. Top row: Projected wireframe. Bottom row: Deformed mesh shaded and seen from a different viewpoint.

Fig. 8 shows the behavior of our algorithm when applied to a more cloth-like material that is more flexible and requires the use of 45 principal components instead of the 30 we used for paper. The deformation is mostly perpendicular to the image plane, which again makes it challenging to track. As can be seen in the video we supply, the reprojected shape closely matches the object in the images, except occasionally near the corners. This

can be attributed to the fact that, because the fabric is very textured, our simple approach to detecting edges can become confused and should be replaced by a more sophisticated one.

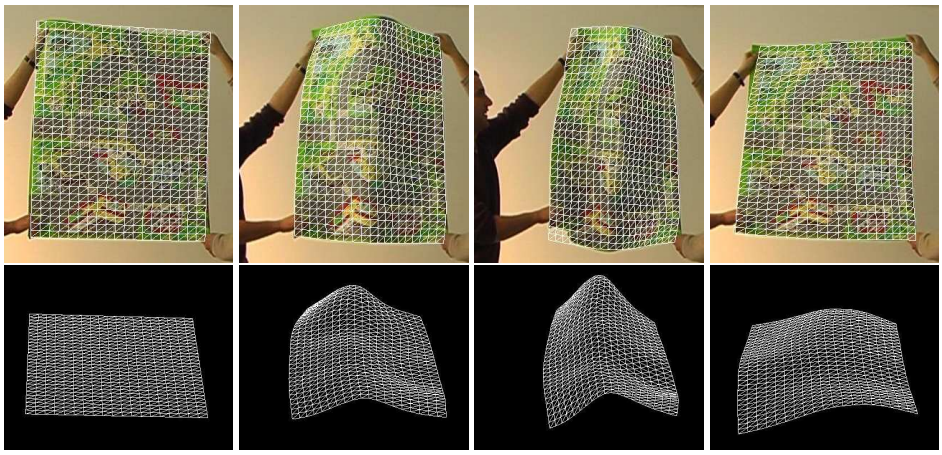


Figure 8: Deforming fabric. The results are displayed in the same manner as in Fig 6. Since the fabric is highly textured, borders of the mesh are sometimes mismatched with texture edges, resulting in small misalignments.

5.2 Stretchable Surfaces

We used an inflating and deflating balloon to test our algorithm's behavior when the surface can stretch or shrink. In all the balloon examples presented here, the initial mesh shapes were obtained by scanning the balloons before starting inflation or deflation and fitting our mesh models to the scans.

All the results shown above involved the use of the penalty term E_D of Eq. 3 to force the mesh edges to retain their original lengths. In Fig. 9, we allow the mesh to stretch by setting the weight of this E_D term to zero. Since we are not tracking the whole balloon, but only its textured part, we only use correspondences and ignore edges. The mesh then expands along with the balloon, which is made possible by principal components such as the one depicted by Fig. 4(b). As shown in Fig. 1(d), the opposite behavior is observed when the balloon deflates.

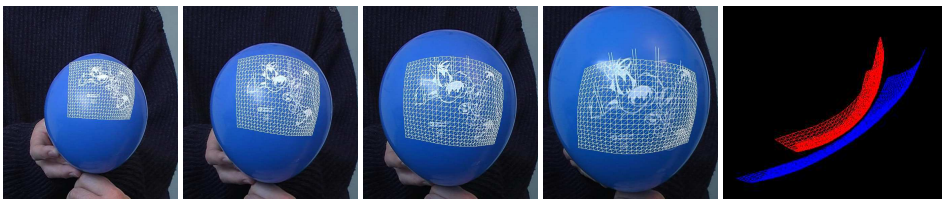


Figure 9: Tracking an inflating balloon with an extensible mesh. Notice the mesh keeps on covering the same portion of the balloon. The last image shows the superposition of the initial mesh in red and the much bigger final one in blue.

6 Conclusion

In this paper, we have presented an approach to parameterizing deformable surfaces of planar topologies in terms of a small number of angles and producing a representative set of shapes by randomly sampling these angles. We have exploited this to create a database of meshes that are naturally registered to one another and are therefore directly amenable to dimensionality reduction using PCA. The resulting low-dimensional models have proved effective for monocular 3–D tracking of surfaces undergoing large deformations.

The set of shapes we produce is however far from linear and PCA may not be the best possible approach to dimensionality reduction. In future work, we therefore intend to explore the use of non-linear techniques to reduce the required number of parameters even further.

References

[1] A. Bartoli and A. Zisserman. Direct Estimation of Non-Rigid Registration. In *British Machine Vision Conference*, Kingston, UK, September 2004.

[2] Michael J. Black and Allan D. Jepson. Eigentracking: Robust matching and tracking of articulated objects using a view-based representation. In *European Conference on Computer Vision*, pages 329–342, 1996.

[3] V. Blanz, C. Basso, T. Poggio, and T. Vetter. Reanimating Faces in Images and Video. In *Eurographics*, Granada, Spain, September 2003.

[4] V. Blanz and T. Vetter. A Morphable Model for The Synthesis of 3–D Faces. In *Computer Graphics, SIGGRAPH Proceedings*, Los Angeles, CA, August 1999.

[5] L. Cohen, , and I. Cohen. Deformable models for 3-d medical images using finite elements and balloons. In *Conference on Computer Vision and Pattern Recognition*, pages 592–598, 1992.

[6] D. Terzopoulos D. Metaxas. Constrained deformable superquadrics and nonrigid motion tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(6):580–591, 1993.

[7] H. Delingette, M. Hebert, and K. Ikeuchi. Deformable surfaces: A free-form shape representation. In *Proc. SPIE Geometric Methods in Computer Vision*, volume 1570, pages 21–30, 1991.

[8] G. Irving, J. Teran, and R. Fedkiw. Invertible finite elements for robust simulation of large deformation. In *SCA '04: Proceedings of the 2004 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 131–140, New York, NY, USA, 2004. ACM Press.

[9] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active Contour Models. *International Journal of Computer Vision*, 1(4):321–331, 1988.

[10] T. McInerney and D. Terzopoulos. A dynamic finite element surface model for segmentation and tracking in multidimensional medical images with application to cardiac 4d image analysis. *Computerized Medical Imaging and Graphics*, 19(1):69–83, 1995.

[11] D. Metaxas and D. Terzopoulos. Shape and Nonrigid Motion Estimation through Physics-Based Synthesis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(6):580–591, 1991.

[12] A. Pentland. Automatic extraction of deformable part models. *International Journal of Computer Vision*, 4(2):107–126, March 1990.

[13] G. Picinbono, H. Delingette, and N. Ayache. Real-time large displacement elasticity for surgery simulation: Non-linear tensor-mass model. In *In Third International Conference on Medical Robotics, Imaging And Computer Assisted Surgery: MICCAI 2000*, pages 643–652, 2000.

[14] H. Sidenbladh, M. J. Black, and D. J. Fleet. Stochastic tracking of 3D human figures using 2D image motion. In *European Conference on Computer Vision*, June 2000.

[15] D. Terzopoulos and D. Metaxas. Dynamic 3D models with local and global deformations: Deformable superquadrics. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13:703–714, 1991.

[16] L.V. Tsap, D.B. Goldgof, S. Sarkar, and W.C. Huang. Efficient nonlinear finite element modeling of nonrigid objects via optimization of mesh models. *CVIU*, 69(3):330–350, March 1998.